In Laube's paper "Finding REMO - Detecting Relative Motion Patterns in Geospatial Lifelines (2004 GIS)", he proposes a geographic data mining approach to detect generic group motion patterns such as flock, convergence, and leader. In the approach, a concept called REMO is declared based on two key features: First, a transformation of motion data to a REMO matrix featuring motion attributes (i.e. speed, change of speed or motion azimuth); second, matching of formalized patterns on the matrix (Figure 1).

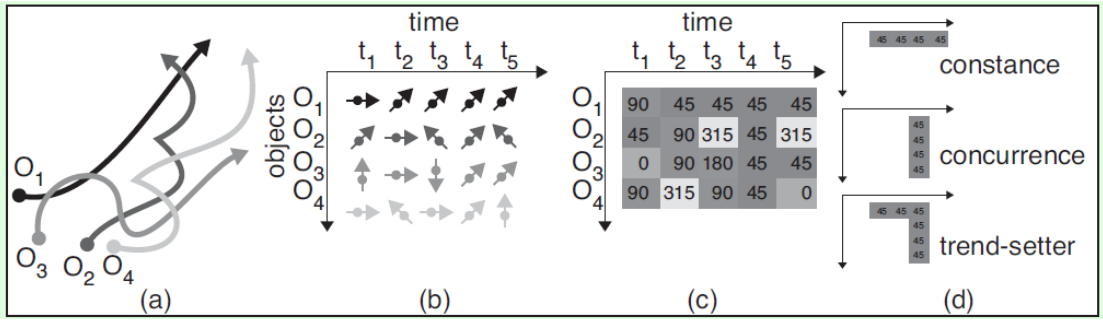

Figure 1 basic motion patterns

Later, the authors describe three basic motion patterns:

- *Constance*: Sequence of equal motion attributes for *r* consecutive time steps (e.g. deer *O1* with motion azimuth 45° from t2 to t5).
- *Concurrence:* Incident of *n* MPO (Motion Point Objects) showing the same motion attributesvalue at time *t* (e.g. deer *O1*, *O2*, *O3*, and *O4* with motion azimuth 45° at *t4*)
- *Trend–setter:* One trend-setting MPO anticipates the motion of *n* others. Thus, a trend-setter consists of a c*onstance* linked to a concurrence (e.g. deer *O1* anticipates at *t2* the motion azimuth 45° that is reproduced by all other MPOs at time *t4*)

To follow Waldo Tobler's first law of geography, near things are more related than distant things. The authors add three spatially constrained REMO patterns (Figure 2):

- *Track:* Consists of the REMO pattern constance and the attachment of spatial constraint. Definition: *constance* + spatial constraint *S*.
- *Flock:* Consists of the REMO pattern concurrence and the attachment of a spatial constraint. Definition: *concurrence* + spatial constraint *S*.
- *Leadership:* Consists of the REMO pattern trend-setter and the attachment of a spatial constraint. For example the followers must lie within the range $(\partial x, \partial y)$ when they join the motion of the trend-setter. Definition: *trend-setter* + spatial constraint *S*.

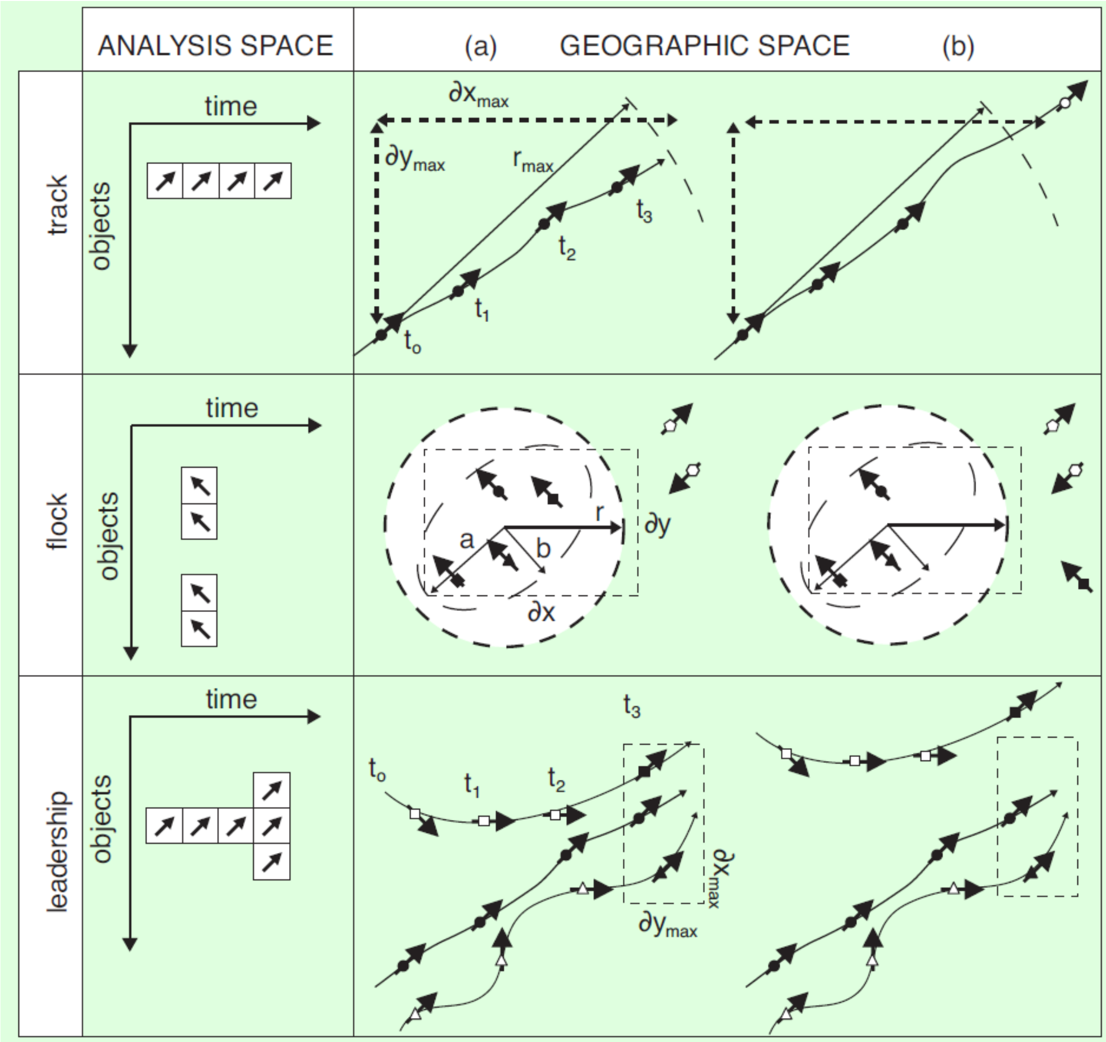

Figure 2 spatially constrained patterns

To answer wildlife biologists' questions about who, when and where of motion patterns, they list another two spatial REMO patterns: convergence and encounter (Figure 3).

- *Convergence:* Heading for *R*. Set of *m* MPOs at interval *i* with motion azimuth vectors intersecting within a range *R* of radius *r*.
- *Encounter:* Extrapolated meeting within R. Set of m MPOs at interval *i* with motion azimuth vectors intersecting within a range R of radius *r* and actually meeting within R extrapolating the current motion.

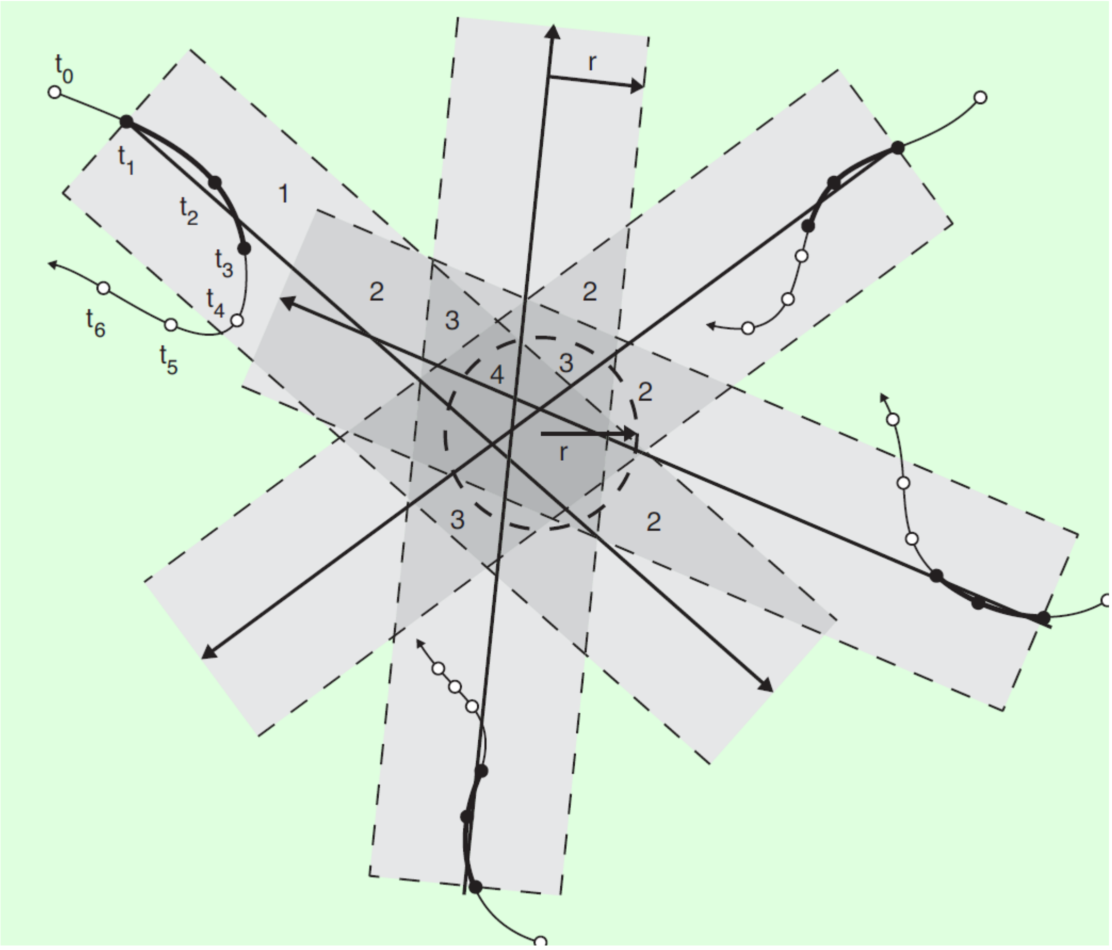

Figure 3 Geometric detection of Convergence. The pattern convergence is found if the darkest polygon exists.

This paper gives a simple description of detecting algorithms, but does not implement the detecting processes.

A few months later, Joachim and his collaborators publish a paper "Efficient Detection of Mothion Patterns in Spatio-Temporal Data Sets". In this paper, they propose approximation algorithms to detect these motion patterns in an approximation way. They list a table to compare the running time bounds for finding patterns as follows.

Pattern	Exact (from [14])	Exact (new)	Approximate
Flock	$O(nm^2 + n \log n)$	-	$O(\frac{n}{\varepsilon^2} \log \frac{1}{\varepsilon} + n \log n)$ (radius)
Leadership	$O(ns + nm^2 + n \log n)$	-	$O(ns + \frac{1}{\varepsilon^2} n \log \frac{1}{\varepsilon} + n \log n)$ (radius)
Convergence	$O(n^2)$	-	$O(n^{2+\delta}/(\varepsilon m))$ (subset)
Encounter	$O(n^4)$	$O(n^3)$ (all) $O((m + \log n)n^2)$ (detect) $O((M + \log n)n^2 \log M)$ (largest)	$O(\frac{1}{\varepsilon} n^2 \log n)$ (radius)

For those detection themes can be applied to search group motion patterns, we have to find some new ideas to build our system. The first idea I have thought is the interface. During the motion detection, each algorithm needs input data and the data should be indicated by users. We can provide an interaction tool to allow users choose their interesting data. However, I am not sure about that a single contribution can be accepted by conferences like SIGMOD. So, I look up all papers about searching that published on SIGMOD. I find that most papers intent to solve some problems and their contributions focus on algorithms and relating performance. From the mentioned performance table, we can see that most detection algorithms need >O(nlogn) running time. If we implement these motion detecting algorithms on Spark, the performance can be improved. And if we change the data model (not REMO), our contributions will be strengthened.

I will think about a new data model suitable for MapReduce.